Configuring Breakpoints

In this section:

Basics

For a breakpoint, you can configure the following properties:

- Actions to be performed upon hitting a certain breakpoint.
- Suspend policy, which defines whether the application should be suspended upon hitting the breakpoint.
- Dependencies on other breakpoints.
- Conditions defining when a breakpoint is hit.

IntelliJ IDEA suggests the following way to change the breakpoints properties:

- Using the Breakpoints dialog box, for a breakpoint selected in the list.

Configuring breakpoints

To configure actions, suspend policy and dependencies of a breakpoint

1. Do one of the following:
   - Right-click a breakpoint in the left gutter, and then click the link **More** or press `Ctrl+Shift+F8`.
   - Open the **Breakpoints** dialog box as described on page Accessing Breakpoint Properties and select the desired breakpoint in the list.
   - In the Favorites tool window, select the desired breakpoint, and click  .

   > Note that the pop-up window shows less options than the Breakpoints dialog box. To show hidden options, click **More**.

2. Define the actions to be performed by IntelliJ IDEA on hitting breakpoint:
   - To display the reaching of a breakpoint as a text message in the debugging console, select the **Log message to console** check check box.

     To evaluate an expression in the context of a breakpoint and display its value in the debugging console, check the option **Log evaluated expression**, and enter a valid expression in the option field.

     This feature lets you obtain information about your running application without having to suspend its execution.
   - To set a breakpoint the current one depends on, select it from the **Depends on** drop-down list.
   - Enable suspending an application upon reaching a breakpoint by selecting the **Suspend** check box, and then select one of the option buttons to specify the way a running program will be paused. For more information on the **Suspend** options, refer to Breakpoints dialog reference.
   - To set the break condition, enable condition by selecting the appropriate check box, and enter the desired expression in the **Condition** field.

     If the expression evaluates to *true*, the user-selected actions are performed. If the evaluation result is *false*, the breakpoint does not produce any effect.

3. The following options are defined in the Breakpoints dialog box (if you edit properties of a particular breakpoint, click **More**):
   - To limit breakpoint hits only with particular object instances using instance IDs, check the **Instance filters** option and type the instance ID value, or click the ellipsis button and specify instance ID in the **Instance Filters** dialog.
   - To define breakpoint behavior with regards to particular classes, select the **Class Filter** check box and specify the class filter. Type the class filter manually or click the **Browse** button  and create the class filter definition in the Class Filters dialog box that opens.
   - To define the number of times a breakpoint is reached but ignored, select the **Pass count** check box and specify the number of passes the breakpoint should be skipped before it is hit.

     For more information, refer to the Breakpoints dialog reference.

See Also

Concepts:

- Run/Debug Configuration
- Breakpoints

Reference:

- Breakpoints

- Favorites Tool Window

Web Resources:

- Developer Community